HSRA: Hindi Stopword Removal Algorithm

Vandana Jha*, Manjunath N*, P Deepa Shenoy* and Venugopal K R*
*Department of Computer Science and Engineering
University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India
Email: vjvandanajha@gmail.com

***Abstract*—In the last few years, electronic documents have been the main source of data in many research areas like Web Mining, Information Retrieval, Artificial Intelligence, Natural Language Processing etc.. Text Processing plays a vital role for processing structured or unstructured data from the web. Preprocessing is the main step in any text processing systems. One significant preprocessing technique is the elimination of functional words, also known as stopwords, which affects the performance of text processing tasks. An efficient stopword removal technique is required in all text processing tasks. In this paper, we are proposing a stopword removal algorithm for Hindi Language which is using the concept of a Deterministic Finite Automata (DFA). A large number of available works on stopword removal techniques are based on dictionary containing stopword lists. Then pattern matching techique is applied and the matched patterns, which is a stopword, is removed from the document. It is a time consuming task as searching process takes a long time. This makes the method inefficient and very expensive. In comparision of that, our algorithm has been tested on 200 documents and achieved 99% accuracy and also time efficient.**

Keywords*—*Artificial Intelligence, Hindi Text Processing, Natural Language Processing, Stopword Removal, Text Processing

I. Introduction

Preprocessing is an important step in the data mining process. It transforms the raw data from original data source to a format that will be more easily and effectively processed further. This preprocessed format is suitable for employing different types of feature extraction methods [1]. It includes cleaning, normalization, transformation, feature extraction and selection etc. Text Processing methods are based on the recognition and derivation of depictive features for documents in natural languages. In order to identify and extract the features, it is necessary to filter out the word or phrases that is not important from the important text [2]. These not so important words, which are evenly distributed in a document, are the most frequent words of any language. These are known as functional words or stopwords. For example, Some of the stopwords are कब (*when*), आप (*you*), आपका (*yours*), करना (*do*). Stopwords have been identified as not important since the earliest days in Text Processing tasks [3]. These words are very frequent and have no additional meaning to the actual content and meaning of a document. These words can be categorized into several word-groups like prepositions, conjunctions, adverbs etc. Removal of these words saves a lot of processing time and memory space and does not damage information retrieval effectiveness [4]. Stopword removal comes under data cleaning part of preprocessing. These removal can reduce text count in the document corpus by 35-45%.

The most of the existing text processing tasks uses the pattern matching method for removal of stopwords. This method uses a dictionary to store stopword lists. The document corpus is searched for the words present in this dictionary and when words are matched, they are removed from the document corpus. This method requires a lot of space to store the dictionary and too much time in searching the word in the document corpus. Our method does the same task of stopword removal in much more efficient way by the implementation of DFA.

978-1-4673-6621-2/16$31.00

A. Motivation

Stopwords are the frequent words in any document corpus with no additional meaning to the content of the text and removal of these words not affect information retrieval from the corpus. Removal of these words can reduce the corpus by 35-45% and this makes text mining methods more efficient so this area is worth exploring. Stopword removal itself consumes a considerable amount of time and it is just a preprocessing step in many text processing tasks so getting this step done in minimum time is a prerequisit for many application areas.

Over the last decade, because of speedy procreation of the Internet, we have acknowledged an accelerated increase of digital documents in Asian languages. Very small amount of work is done in Asian languages. Although TREC conferences gave lot of space for Chinese language, most of the results cannot be imported to other Asian languages as they differ a lot linguistically. Our algorithm is for Hindi language. Hindi, the 4th largest spoken language, is the official language of India. It has 490 million speakers across the world, which is 4.7% of the world population [5]. Only 28.6 percent of Internet users understand English [6] so it is important to focus on other languages. But it is an uphill task for a resource scarce language like Hindi.

B. Contribution

In this paper, a stopword removal algorithm is proposed which is based on DFA implementation. It takes 200 Hindi documents as an input which is movie review data set collected from *http://hindi.webdunia.com/bollywood-movie-review* [7] and returns the documents with stopwords removed from them. For DFA implementation, it uses json objects which takes care of current state, accepting state, start state, final state, transitions and input characters. Our algorithm takes 1.77 seconds to remove stopwords from tested documents whereas existing pattern matching method takes 3.4 seconds to do the same task under similar environments.

C. Organization

The paper is organized as follows: Section II provides a brief overview of the related work. Section III describes stopword removal algorithm beased on DFA for Hindi language. State transition tables and the related results are discussed in section IV. Section V describe conclusions of the paper.

II. Related Work

In the past, there exists some research works for stopwords removal.
Paper [8] created a Chinese stopword list of 1289 words. They obtained this by merging the classical stopword list with the stopwords of different domains. Their experiment pointed out that the hash-filter method was the fastest for stopwords removal. Zou et al. [9] work was for Chinese language. They proposed an automatic aggregated methodology for measuring the word frequency characteristic by statistical model and its information characteristic by information model. Their method was based on the perception that stopwords are most frequent words and also they have a balanced allocation in the documents. Based on this idea they redefined that stopwords are the words which are evenly distributed and high occurring in documents. Their developed stopword list was comparable and was much more general than other existing Chinese stopword lists. They constructed a standard and displayed significant progress. Hao and Lizhu [10] defined a stopword list in Chinese language which was based on weighted Chi-squared statistic. Association rule mining using Genetic Algorithm is used in the papers [11], [12], [13].

In [14] an algorithm for removing stopwords have been proposed which is based on a finite state machine in Arabic language. A hybrid stopword removal technique was implemented based on a dictionary and an algorithm for Arabic language [15]. A set of 242 abstracts selected from the Proceedings of The Saudi Arabian National Computer conferences and another set of data selected from the Jordanian Alrai Newspaper were used to test their approach. In [16] a corpus-based list from newswire and query sets and a general list using the same corpus have been constructed and then comparision was done for the effectiveness of these lists on the information retrieval systems. Three stopword lists were compared and these were general stop-list, corpus based stop-list and a combined stop-list. The general stop-list was generated using only Arabic language structure characteristics and has 1377 words. This list was without any additions. Second list, the corpus based stop-list has 359 words. It was generated by the words whose occurrence is more than 25,000 times in the corpus. Third list was the combination of above mentioned two stop-list and has 1529 words. Their results showed that first list performed better than the other two lists. In [17] a stopword list have been generated in Arabic language.

Savoy [18] defined a stopword list for French language. In their method, they sorted the words based on their occurrence frequency in their corpora and filtered out 200 frequently occurring words. They also removed all numeric values (e.g., "2001", "5"), all adjectives and nouns and added some non-informative words. Examples of some of the words, they added are, several possessive or personal pronouns like "tien" (*yours*), "moi" (*me*), conjunctions ("cependant" (*however*)) and prepositions ("dessus" (*upon*)). Their given French stopword list has 215 words. According to Myerson [19], a word should satisfy two conditions to be a stopword. First condition was, it should be a highly frequent word in a document. Second condition was, the statistical correlations with all the classification categories should be small. The statistical correlation between a word and classification categories was measured using weighted Chi-squared statistic. Their results showed that the stopword list involving constructed 500 words can reduce the corpus size by 43%. Zheng and Gaowa [20] gave a method for constructing stopword list for Mongolian language which was based on entropy calculation. In this method, entropy of every word was calculated and then the words were sorted according to that. This was the initial stopword list. For final stopword list, initial list was combined with the Mongolian part of speech. Sinka and Corne [21] generated stopword list based on entropy. In this method, three stopwords list were generated. First list was generated from random web pages, second was generated from the BankSearch dataset and third was from unsupervised clustering experiments. Their experimental results showed that the generated stopword lists were outperformed the existing stopword lists, mostly in the case of hard classification tasks.

In [22], the authors investigated the effect of stopword removal and stemming on Hindi Information Retrieval. In this study, three different stemmers have been used and their performance has been compared. The data set for the experiments were a test collection generated using Hindi documents from EMILLE corpus. They concluded that retrieval improves considerably after stopword removal. In [23], the authors investigated the impact of stemming, stop word removal and size of context window on Hindi word sense disambiguation. The simulations have been carried out on a manually created sense tagged corpus containing Hindi words (nouns). Hindi Wordnet has been provided the sense definition. It is a significant lexical resource developed at IIT Bombay for Hindi language. After stemming and stopwords removal, the maximum observed precision of 54.81% on 1248 test instances were noted. Paper [24] proposed a multi-domain sentiment aware dictionary. Paper [25] performed subjectivity analysis at the sentence level. They achieved 71.4% agreement with human annotators and 80% accuracy in classification on a parallel data set in English and Hindi.

III. Proposed Work

Main Algorithm describes the overview of the proposed work.

The document containing Hindi text is parsed at sentence level and for each sentence in the document, it is parsed at word level. For each word, DFA_STOPWORD() function is called. The output returned by the function is checked for true or false, boolean values. If it returns false then the word is not a stopword and appended in the document. If it returns true then the word is a stopword and removed from the document.

Function DFA_STOPWORD() illustrates a faster method for stopword removal using DFA implementation. It considers json object which acts as input to algorithm. Json object consists of five parameters namely *states*, *character*, *transitions*,

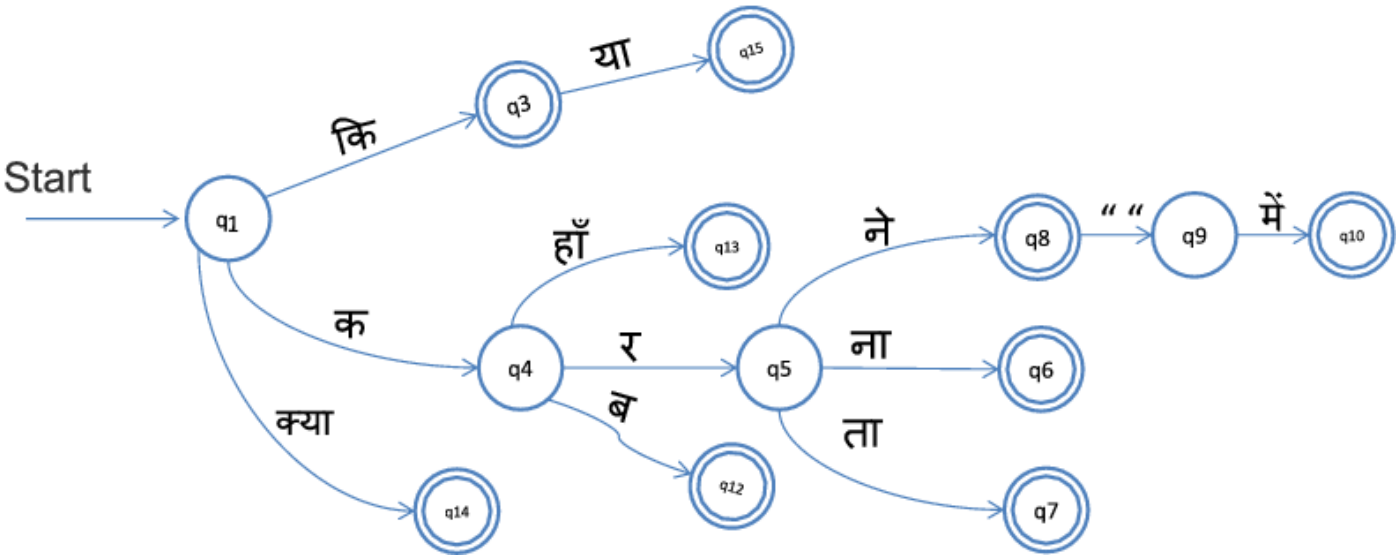

Fig. 1: A sample of DFA showing transition states

Main Algorithm: Stopword Removal Algorithm

Data: Document containing Hindi text
Result: Document with stopwords removed

```
begin
    Initialize:
    words_checked = " "
    Perform:
    for each d in document do
        for each word in d do
            result = call DFA_STOPWORD(word)
            if result == false then
                //append because its not a stop word
                words_checked.append(word)
            end
        end
        return words_checked
    end
end
```

Function: DFA_STOPWORD

Data: DFA designed json object and tokenized word
Result: returns true if stopword else returns false

```
begin
    Initialize:
    states, character, transitions, start_state,
    accepting_state
    current_state = start_state
    Perform:
    after_splitting = call split() // to tokenize the word
    along with matra's
    for each character in after_splitting do
        if current_state is not in transitions then
            return false // invalid state, not defined in
            DFA states
        end
        if character is not in transitions[current_state]
        then
            return false // there is no transition on such
            inputs, marking it as invalid
            transit to the respective state and update the
            current_state to new_state
        end
        if current_state in accepting_state then
            return true
        else
            return false
        end
    end
end
```

start_state and *accepting_state*. *Character* is Hindi character set. *Transitions* define the next state upon each input character read. *Start_state* indicates the starting of DFA. *Accepting_state* double circled ensures DFA accepts the input. Split function is called, to tokenize the word along with matra's. For example, if the word is "आपका" then it can be split into आ + प + का. This will help in traversing the word. For each *character*, initial conditions are checked. If *current_state* is not in defined *transitions* then it returns false. At any point of time, if DFA encounters a *character* that is not in *transitions* then it returns false. After all the above mentioned conditions are met, DFA transits to the respective state and updates the *current_state* to *new_state*. If *current_state* is an *accepting_state* then DFA returns true else returns false. True indicates that the word is a stopword and false indicates that it is not a stopword.

IV. Simulation Results

Table I shows a sample of the state table which is used by json objects to implement DFA method for stopword removal. Here, state "1" is acting as start state and shown by start -> symbol and then the transitions are shown by the cells in the table. Some cells are shown filled with the numbers and these

TABLE I: Sample of State Transition Table

	states	क	हाँ	ब	कि	या	र	क्या	ना	ता	ने	में	“ ”
start –>	1	4			3			14					
	2												
*	3					15							
	4		13	12			5						
	5								6	7	8		
*	6												
*	7												
*	8												9
	9											10	
*	10												
	11												
*	12												
*	13												
*	14												
*	15												

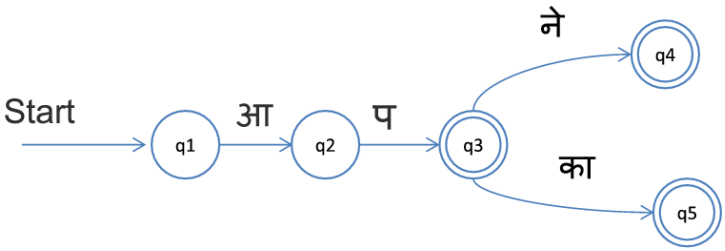

Fig. 2: A small sample of DFA showing states starting from आ

numbers denote the next state for the transition. Some cells are shown empty and these cells can be utilized further for more extended DFA. State preceded with the cell containing “*” denotes accepting states. Fig. 2 shows a small DFA showing the states starting with “आ”. A slightly better view of DFA is available in Fig. 1.

V. Conclusions

In this paper, a stopword removal algorithm is implemented for Hindi Language which is based on DFA. It takes Hindi documents as an input and returns the document with stopwords removed from them. Most of the existing stopword removal techniques uses the pattern matching method for removal of stopwords. Pattern matching method uses a dictionary to store stopword lists. The document corpus is searched for the words present in this dictionary and when words are matched, they are removed from the document corpus. This process requires a lot of space to store the dictionary and too much time in searching the word in the document corpus. Because of this, the method becomes inefficient and very expensive. Our method does the same task of stopword removal but it takes 1.77 seconds to remove stopwords from tested 200 documents with achieved accuracy of 99% whereas existing pattern matching method takes 3.4 seconds under similar environments. These results make our method more time and space efficient. In future, we are planning to build another time and space efficient system for stemming and for other preprocessing tasks.

References

[1] R. Feldman and J. Sanger, “The text mining handbook.”

[2] G. Miner, R. Nisbet, and J. Elder IV, *Handbook of statistical analysis and data mining applications*. Academic Press, 2009.

[3] H. P. Luhn, “A statistical approach to mechanized encoding and searching of literary information,” *IBM Journal of research and development*, vol. 1, no. 4, pp. 309–317, 1957.

[4] B. F. William and R. Baeza-Yates, “Information retrieval: Data structures and algorithms,” *ISBN-10*, vol. 134638379, 1992.

[5] [Online]. Available: http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

[6] (2013, December 31). [Online]. Available: http://www.internetworldstats.com/stats7.htm

[7] V. Jha, N. Manjunath, P. D. Shenoy, K. Venugopal, and L. Patnaik, “Homs: Hindi opinion mining system,” in *Recent Trends in Information Systems (ReTIS), 2015 IEEE 2nd International Conference on*. IEEE, 2015, pp. 366–371.

[8] Z. Yao and C. Ze-wen, “Research on the construction and filter method of stop-word list in text preprocessing,” in *Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on*, vol. 1. IEEE, 2011, pp. 217–221.

[9] F. Zou, F. L. Wang, X. Deng, S. Han, and L. S. Wang, “Automatic construction of chinese stop word list,” in *Proceedings of the 5th WSEAS international conference on Applied computer science*, 2006, pp. 1010–1015.

[10] L. Hao and L. Hao, “Automatic identification of stop words in chinese text classification,” in *Computer Science and Software Engineering, 2008 International Conference on*, vol. 1. IEEE, 2008, pp. 718–722.

[11] P. D. Shenoy, K. Srinivasa, K. Venugopal, and L. M. Patnaik, “Evolutionary approach for mining association rules on dynamic databases,” in *Advances in knowledge discovery and data mining*. Springer, 2003, pp. 325–336.

[12] P. D. Shenoy, K. Srinivasa, K. Venugopal, and L. M. Patnaik, “Dynamic association rule mining using genetic algorithms,” *Intelligent Data Analysis*, vol. 9, no. 5, pp. 439–453, 2005.

[13] V. H. Bhat, P. G. Rao, R. Abhilash, P. D. Shenoy, K. Venugopal, and L. Patnaik, “A data mining approach for data generation and analysis for digital forensic application,” *IACSIT International Journal of Engineering and Technology*, vol. 2, no. 3, pp. 314–319, 2010.

[14] R. Al-Shalabi, G. Kanaan, J. M. Jaam, A. Hasnah, and E. Hilat, “Stop-word removal algorithm for arabic language,” in *Proceedings of 1st International Conference on Information & Communication Technologies: from Theory to Applications, CTTA’04*, 2004, pp. 545–550.

[15] B. Alhadidi and M. Alwedyan, “Hybrid stop-word removal technique

for arabic language." *Egyptian Computer Science Journal*, vol. 30, no. 1, pp. 35–38, 2008.

[16] I. A. El-Khair, "Effects of stop words elimination for arabic information retrieval: a comparative study," *International Journal of Computing & Information Sciences*, vol. 4, no. 3, pp. 119–133, 2006.

[17] A. Alajmi, E. Saad, and R. Darwish, "Toward an arabic stop-words list generation," *International Journal of Computer Applications*, vol. 46, no. 8, 2012.

[18] J. Savoy, "A stemming procedure and stopword list for general french corpora," *JASIS*, vol. 50, no. 10, pp. 944–952, 1999.

[19] R. B. Myerson, *Fundamentals of social choice theory*. NorthWestern University, Center for Mathematical Studies in Economics and Management Sciences, 1996.

[20] G. Zheng and G. Gaowa, "The selection of mongolian stop words," in *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, vol. 2. IEEE, 2010, pp. 71–74.

[21] M. P. Sinka and D. W. Corne, "Towards modernised and web-specific stoplists for web document analysis," in *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on*. IEEE, 2003, pp. 396–402.

[22] A. K. Pandey and T. J. Siddiqui, "Evaluating effect of stemming and stop-word removal on hindi text retrieval," in *Proceedings of the First International Conference on Intelligent Human Computer Interaction*. Springer, 2009, pp. 316–326.

[23] S. Singh and T. J. Siddiqui, "Evaluating effect of context window size, stemming and stop word removal on hindi word sense disambiguation," in *Information Retrieval & Knowledge Management (CAMP), 2012 International Conference on*. IEEE, 2012, pp. 1–5.

[24] V. Jha, S. R, S. Hebbar, P. D. Shenoy, and V. Kuppanna Rajuk, "HMD-SAD: Hindi Multi-Domain Sentiment Aware Dictionary," in *2015 IEEE International Conference on Computing and Network Communications (CoCoNet) (CoCoNet'15)*, Trivandrum, Kerala, India, India, Dec. 2015, pp. 247–253.

[25] V. Jha, N. Manjunath, P. D. Shenoy, and V. Kuppanna Rajuk, "HSAS: Hindi Subjectivity Analysis System," in *2015 Annual IEEE India Conference (INDICON) (IEEE INDICON 2015)*, Jamia Millia Islamia, New Delhi, India, 2015.